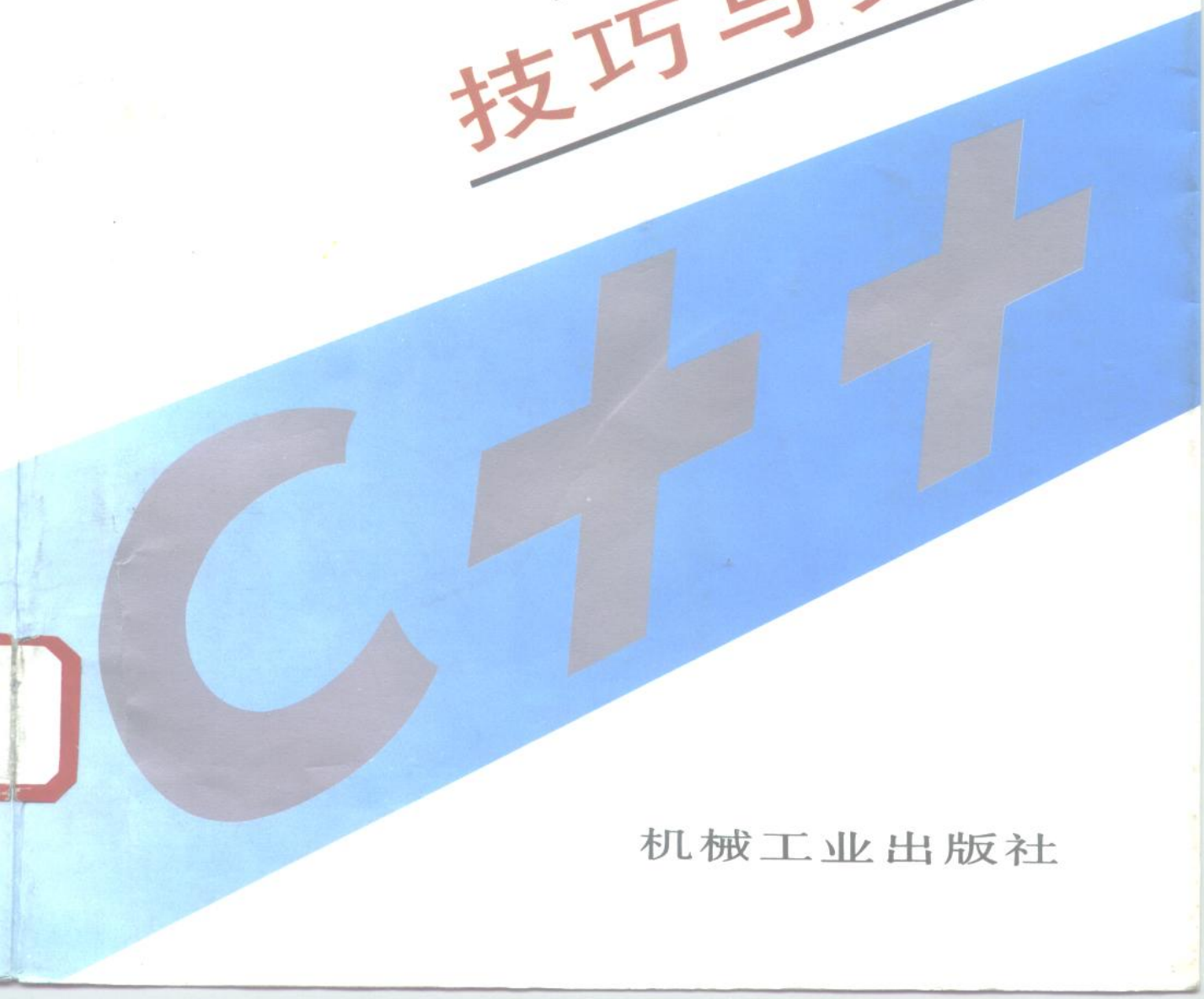
顾仁　等编著

高级C++语言程序设计技巧与实例



机械工业出版社

高级 C++语言程序设计技巧与实例

顾仁　等编著

机 械 工 业 出 版 社

北京　1995

C++ 语言是开发计算机系统软件和应用软件的常用工具，其特点是效率高、代码简洁、功能全面、可移植性好，而且具有数据封装和继承特性，使程序代码可以重复使用。

本书由浅入深地系统介绍了 C++ 语言的入门与应用。全书分成两篇：第 1 篇是基本技能训练，包括近 200 个上机练习实例，内容涉及 I/O、程序流程控制、函数、变量存储类型、类与对象、数组、重载操作符、继承、虚拟函数、友元函数、this 指针、类库及鼠标控制等；第 2 篇是高级绘图技巧，包括近 80 个练习实例，内容涉及 BGI 字体与文本、图形设计技术、图标、窗口、动画、交互式绘图工具的制作以及 CAD 程序设计。

本书内容比较全面，程序实例丰富，可供大中专院校师生及 C++ (包括 Borland C++ 和 Turbo C++) 程序员参考，也可作为上机培训教材使用。

图书在版编目（CIP）数据

高级C++语言程序设计技巧与实例/顾江等编著. - 北京：机械工业出版社，1995

ISBN 7-111-04822-9

Ⅰ.高… Ⅱ.顾… Ⅲ.C 语言、C++—程序设计 Ⅳ.TP312C

中国版本图书馆CIP数据核字（95）第11886号

出 版 人：马九荣 （北京市百万庄南街1号 邮政编码100037）
责任编辑：王中玉 版式设计：霍永明 责任校对：肖新民
封面设计：方 芬 责任印制：路 琳
北京市房山区印刷厂印刷 新华书店北京发行所发行
1995年12月第1版第1次印刷
787mm×1092mm 1/16 29.5印张 721千字
0 001—3 000册
定价：42.00元

前　言

C++ 语言是开发计算机系统软件和应用软件的常用工具，原因在于 C++ 语言效率较高、代码简洁、功能全面、可移植性好，而且具有数据封装和继承特性，使得程序代码可重复使用。现在，市面上比较流行的 C++ 语言版本有 Turbo C++，Borland C++，Microsoft C／C++ 和 Visual C++，其中后三种版本不仅支持传统 DOS 程序设计，还支持 Windows 程序设计。

目前，市面上已有不少关于 C++语言的书籍，但大多数是先介绍 C 语言，然后介绍 C++语言，而且在讨论 C++语言时侧重于描述它的语法。本书将略去 C 语言的有关内容，而直接讨论 C++程序设计技巧，并给出了大量程序实例。

本书中的程序采用 Turbo C++和 Borland C++(两者兼容)编写而成，并侧重于 DOS 程序设计。有关 C 语言和 Windows 程序设计的内容，请读者参考其他相关书籍。程序实例将贯穿全书，旨在让读者根据实例体会有关技巧，并通过上机实习掌握复杂的程序开发方法。

值得指出的是，本书所提供的程序实例都适用于虚拟图形阵列（VGA）、增强型图形适配器（EGA）、彩色图形适配器（CGA）及 Hercules 图形适配器。正是这种灵活性，把它们移植到读者的计算机系统（包括 IBM PC 及其兼容机）上运行是很容易的。

本书由浅入深地系统讨论了 C++语言的入门与应用，全书分成两篇：第 1 篇是基本技能训练，包括第 1 章到第 13 章。这一篇包含近 200 个上机练习实例，内容涉及 I／O、程序流程控制、函数、变量存储类型、类与对象、数组、重载操作符、继承、虚拟函数、友元函数、this 指针、类库及鼠标控制等，这些实例可以帮助读者澄清 C++语言中的一些概念。第 2 篇是高级绘图技巧，包括第 14 章到第 24 章。这一篇包含近 80 个练习实例，内容涉及 BGI 字体与文本、图形程序设计技术、图标、窗口、动画、交互式绘图工具的制作以及 CAD 程序设计，这部分实例属高级训练，由一些具有真正功能的实用程序组成。

本书内容比较全面，程序实例丰富，可供大中专院校师生及 C++ (包括 Borland C++ 和 Turbo C++) 程序员参考，也可作为上机培训教材使用。

参与本书编写的有顾仁、刘民叶、胡希年、方兵、李汉生、叶青青、刘勇、谷贵明、方衡。刘艳波绘制了书中所有的图形。

顾　仁

1995 年

目　录

第 2 篇　高级绘图技巧

第 1 篇　基本技能训练

第1章 简 介

1.1 C与C++

众所周知，C语言是一种容易维护而且功能极强的程序设计语言，所以已成为开发计算机系统软件或数据库软件的重要语言。但是，当系统规模较大，程序代码多于数千行时，变量和函数的维护就成了一个巨大的负担。在这种情况下，具备代码可重用性、可扩充性，并能对数据或模块进行封装保护的"面向对象程序设计(Object Oriented Programming，即OOP)"的概念，就自然受到绝大多数程序设计人员的欢迎。

1980年，贝尔实验室开发出C++语言，其中包含大部分C语言及众多C语言程序库，并具备面向对象程序设计的方法。

OOP提供了新的程序开发概念，包括数据隐藏、封装、操作符和函数重载等。1988年，美国Borland公司集中美国各地编译程序专家的智慧，设计出了能与ANSI C及ANSI C++ 2.0兼容的Turbo C++程序设计语言。

1992年，美国Microsoft软件公司推出第三代窗口操作系统Windows 3.1，该系统提供了良好的用户界面，直接在此环境中运行的C++编译程序是MS C/C++ 7.0和Visual C++，它们都具有OOP功能，而且完全支持Windows，后者还提供了丰富的类函数库MFC。

Borland公司于1992年开发出与Turbo C++类似的编译环境Borland C++ 3.1，它除了具备Turbo C++的功能外，还为专业软件开发人员提供了增强的软件开发工具，它是有助于开发Windows操作系统的程序环境，这些开发工具包括汇编器(assembler)、增强的调试器(debugger)及其他实用程序。现在，Borland公司又推出了Borland C++ 4.0，这无疑会给它的新老用户提供更方便、更有效的编程环境。

1.2 Turbo C++与Borland C++

Turbo C++和Borland C++包含Turbo C 2.0的所有功能，用Turbo C 2.0开发的程序能在Turbo C++中编译和运行。Turbo C++和Borland C++另行扩充和增强的部分包括:

(1) 强化的语法：包括函数引用调用(call by reference)、引用参数(reference argument)、函数声明时缺省参数(default argument)、简单易用的I/O语句、增强的内存分配命令NEW等。

(2) 面向对象程序设计(OOP)技术：包括类定义(class)、友元函数(friend function)、构造函数(constructor)、析构函数(destructor)、虚拟函数(virtual function)、虚拟类(virtual class)、继承(inheritance)、操作符重载(operator overloading)、函数重载

(function overloading)、抽象数据类型(abstract data type)、面向对象的文件和 I／O 流(I／O stream)以及各种类库(class library)。

Borland C++还包括:

(1) 调试器(debugger): 允许程序设计者跟踪程序设计的过程和各类对象的内容等。

(2) 剖析器(profiler): 剖析程序设计的效率。

(3) Windows 3.1 Toolkit(工具箱): 充分支持用户界面的强化。

Turbo C++和 Borland C++可以在 PC／XT, PC／AT, PS／2 系列, 386, 486 及其他兼容机上运行。Turbo C++至少必须在 MS-DOS 2.0 以上, 而 Borland C++至少必须在 MS-DOS 3.0 以上的环境中运行。

Turbo C++和 Borland C++至少必须在内存为 640KB 的环境中运行(同时要求硬盘空间约为 6MB)。它们使用 Small 内存模式, 在 A:盘上存放 TC.EXE, 在 B:磁盘上存放头文件(head file)及程序库(library)。另外, 在至少 1MB 的 RAM 中开设一个虚盘, 以存放程序和可执行文件。Turbo C++亦可在无硬盘但带两个 1.2MB 软盘驱动器及 1MB RAM 的环境中运行。

1.3 安装 Borland C++

Borland C++软件包含有两种 Borland C++编译器的不同版本: 集成环境版本(IDE)和 DOS 命令行版本。它还包括作为 Windows 应用程序的 Turbo C++ for Windows。如果不会使用 DOS 命令, 则在安装 Borland C++之前请先阅读 DOS 参考手册。

Borland C++中有一个称为 INSTALL 的自动安装程序, 必须使用 INSTALL 程序安装 Borland C++。由于使用了文件压缩技术, 所以读者必须使用该安装程序, 不能仅将 Borland C++文件拷贝到硬盘上。INSTALL 自动将被压缩的 Borland C++和 Turbo C++ for Windows 文件恢复过来。为参考起见, README 文件放在含有一系列系统文件的安装盘上。没有拷贝保护。

假设读者已熟知 DOS 命令。例如, 读者知道用 DISKCOPY 命令去备份系统盘。

1.3.1 安装 Borland C++的步骤

建议读者在安装前先阅读一下 README 文件。

INSTALL 程序检查所使用的硬件并且适当地配置 Borland C++, 它也能随需要创建目录并从系统盘(读者购买的盘)上把文件拷贝到硬盘上, 它的操作是自动进行的。下列内容告诉读者所需要知道的信息:

为了安装 Borland C++:

(1) 插入安装盘(disk1)到驱动器 A:, 键入下列命令, 然后按 Enter 键:

```
A: INSTALL
```

(2) 当处于安装屏幕时, 按回车键。

(3) 按照提示操作。

(4) 安装过程完毕后, 将下列行添加到 CONFIG.SYS 文件中:

```
FILES=20
```

将下列行添加到 AUTOEXEC.BAT 文件中(或修改 AUTOEXEC.BAT 中已有的 PATH 语句):

```
PATH=C:\BORLANDC\BIN
```

注意，README 文件含有最新的有关 Borland C++的重要信息。HELPME.DOC 文件也回答一些类似的技术服务问题。

当下一次启动 Microsoft Windows(在从浏览 README 文件的过程退出之后)时，Windows 将创建一个 Borland C++程序组，并将其安装到 Program Manager(程序管理员)中。该程序组包含下列 Borland C++程序和工具图标:

```
Borland C++
Turbo Profiler
Turbo Debugger for Windows
Turbo C++ for Windows
Resource Workshop
WinSight
Import Librarian
Fconvert Utility
```

注意，INSTALL 假设 Microsoft Windows 已安装在读者安装 Borland C++时所指定的 Windows 目录中，它还假设在启动 Windows 时，Program Manager 作为 Windows 的"shell"(外壳)自动启动。如果希望从 Program Manager 中使用一个不同的命令外壳，就应当编辑 Windows 所在目录下的 SYSTEM.INT 文件，向其中加入下列行:

```
SHELL=PROGRAM.EXE
```

否则，在第一次启动 Windows 并且 Borland C++试图创建一个新的 Program Manager 组时，系统将给出提示信息:

```
Cannot Communicate with Program Manager
```

在将一组 Turbo C++ for Windows 及其他工具安装到 Program Manager 组中后，可以检查其设置情况，然后将其重新安装到自己所使用的组中。

1.3.2 保护模式和内存管理

Borland C++使用 DPMI(DOS Protected Mode Interface，即 DOS 保护模式接口)在保护模式下运行其编译器，使得读者不需交换即可访问计算机的全部内存。保护模式接口对读者是完全透明的，除了一些例外的情况（不需考虑它们的存在）之外。

(1) DPMIINST: 例外之一可能出现在第一次运行 Borland C++时。Borland C++使用一个具有多种机器特性的内部数据库来决定如何在用户计算机上启动保护模式，并相应地设置其自身的配置项。如果 Borland C++无法识别用户计算机，用户将被告知如下信息:

```
Machine not in Database (RUN DPMIINST)
```

如果收到上述信息，只需在 DOS 提示符下键入 DPMIINST 以运行 DPMIINST，并按程序的提示进行操作。

DPMIINST 进行一系列测试，以找出启动保护模式的最佳方法，并相应地自动设置 Borland C++的配置项。一旦运行了 DPMIINST 后，就不必再运行它了。

(2) DPMIMEM: 在缺省条件下，Borland C++的 DPMI 接口将为自己分配所有扩展

和扩充内存。如果不希望所有可用内存都被 DPMI 核心所占用，就必须设置一个环境变量，以说明要使用内存的最大值。该变量可直接在 DOS 提示符下输入或作为一行插入到 AUTOEXEC.BAT 文件中。其语法为:

```
DPMIMEM=MAXMEM nnnn
```

其中 nnnn 是以 KB 为单位的内存大小。例如，假设计算机系统有 4MB 内存，且希望 DPMI 核心使用 2MB 空间，而将另 2MB 留出来，那么 DPMIMEM 变量将如下设置:

```
C: > set DPMIMEM = MAXMEM 2000
```

当在 386 增强模式下运行 Windows 3.x 时，不需要设置 DPMIMEM 变量；相反，可以使用一个 Windows PIF 文件来设置 Borland C++的内存使用配置项。

在 Windows 标准模式下，建议在运行 Windows 以前预先装入 Borland DPMI 核心。这是通过运行 DPMIRES.EXE(参阅下面的内容)实现的。当与 Windows 一起使用 DPMIRES 时，必须确保 DPMIMEM 变量的值小于可用最大内存，以保证 Windows 有足够的物理空间运行。

DPMIRES 是一个可与 Borland C++ 3.x 一起使用的 Borland 实用程序。它可在某些情况下提高一些 Borland 语言工具的性能，特别是提高下列工具的性能: BCC，TASMX 和 TLINK。在运行时，DPMIRES 将启动 DOS 保护模式(DOS Protected Mode)接口，并产生一个 DOS 命令外壳。上述应用程序可被较快地装入这个命令外壳。在此外壳下键入 "EXIT"命令可退出该外壳。

当用 MAKER(实模式下的 MAKE)或批处理文件而不是使用保护模式下的 MAKE 进行编译时，DPMIRES 特别有用。在这种情况下，系统将高效地运行 DPMIRES，然后运行 MAKER 或批处理文件，因为编译器每次装入文件的速度加快了。

注意: 在运行 DPMIRES 后，就不能再以增强模式运行 Windows 了。如果需要以增强模式运行 Windows 3.x，必须先退到 DOS 下，再运行 Windows 3.x。

1.3.3 扩展和扩充内存

一旦装入 DPMI 核心(无论是通过 BC 还是通过运行 DPMIRES 实用程序)，Borland C++集成开发环境(IDE)都直接与 DPMI 接口交互作用来分配内存，从而装入文件或进行内存操作。在缺省条件下，IDE 将使用所有 DPMI 核心保留的扩展内存和所有可用的扩充内存(EMS)，EMS 内存作为数据交换缓冲区使用。

选项 Options | Environment | Startup...dialog 和命令行选项 / X 及 / E 可用来改变上述行为。这些选项设置并不影响由 DPMI 核心保留的内存，它们仅影响 IDE 使用的内存量。

Use Entended Memory 对话框(和 / X 命令行选项)可用来通知 BC 还有多少被 DPMI 核心保留的内存是可用的。限制 BC 所能使用的核心内存量的主要原因是为了让其他 DPMI 应用程序能在 IDE 中或在 IDE 打开的 DOS 外壳中运行。

Use EMS Memory 对话框(和 / E 命令行选项)可用来通知 IDE 有多少页的 EMS(每页为 16KB)可作为数据交换缓冲区。除非核心被指示留出一些空闲内存，否则就没有 EMS 页可供 IDE 使用。

1.4 运行 BC

如果安装好 Borland C++后想急于运行它，则可进入 Borland C++的\BIN 子目录中，键入 BC 并按回车键。如果想运行 Turbo C++ for Windows，可在 Program Manager 中将鼠标光标置于 Turbo C++ for Windows 图标上并双击鼠标左按钮。

在试用过 IDE 后，读者也许想将 IDE 的一些选项用户化。IDE 的 Options | Environment | Startup 和 Options | Environment | Colors 选项使这项工作易如反掌。

1.5 膝上计算机系统

如果有一套膝上计算机(具有 LCD 或等离子显示器)，那么除了执行前几节所给出的操作外，在使用 Borland C++之前还要设置屏幕参数。在运行 Borland C++之前，要使 Borland C++的集成开发环境(BC.EXE)正常工作，必须在 DOS 命令行键入 MODE BW80。

虽然可以从 Video 选项组创建一个批处理文件完成这项工作，但也可很容易地用 Options | Environment | Startup 选项在 IDE 中为黑白显示器安装 Borland C++，选择"Black and White / LCD"即可。

1.6 其他

本节汇总了一些琐碎的说明:

(1) README 文件: 当运行 INSTALL 程序后，Borland C++自己显示 README 文件。如果以后再想阅读 Borland C++的 README 文件，可以在 DOS 命令行键入如下命令:

```
README
```

(2) HELPME!.DOC 文件: 安装盘上还包含一个 FILELIST.DOC 文件，它列出了安装盘上的所有文件名及每个文件包含内容的简要说明。其中 HELPME!.DOC 文件包含对通常会遇到的问题的解答，当遇到困难时就使用它。可以使用 README 程序来阅读 HELPME!.DOC 文件。可在命令行键入

```
README HELPME!.DOC
```

(3) 例程: 在 Borland C++软件包中包含大量面向 DOS 和 Windows 的 C++例程的源代码，其中有一个用于展开的称为 Turbo Calc 的源程序代码。这些程序都由 INSTALL 程序安装在..\EXAMPLE 目录下。..\EXAMPLE 目录还含有一些子目录，这些子目录中存放着 Borland C++提供的其他工具和实用程序(如 Turbo Assembler，Turbo Debugger 和 Resource Workshop)范例。在编译这些范例程序前，应先阅读为它们准备的打印文档或联机文档。

(4) 用户化 IDE: Borland C++ 3.x 允许 IDE 内的选项设置完全用户化，这可通过修改出现在 Options | Environment 菜单中的各种选项来完成。这些选项允许读者指定视频模

式、编辑模式、菜单颜色和缺省目录等。

1.7 面向对象的方法

用面向对象(object oriented)的方法解决问题的基本概念是将数据以及处理这些数据的函数结合在一个单元中，这样的单元称为对象(object)。规范化的对象中，包含变量和函数的那些数据类型称为类(class)，因此对象可以看成是类的成员(member of class)，也称为类变量。

对象中的每个函数都称为成员函数(member functions)，它们仅用来存取对象中的成员变量(member variables)。程序设计者不能直接存取对象中的数据，而必须通过成员函数来存取，以避免无意中改变变量的值。这种保护对象数据的方法称为数据隐藏(data hiding)。数据和函数结合在一个单元(对象)中称为封装(encapsulated)。

Turbo C++和 Borland C++中的所谓成员函数在其他面向对象语言(如 Smalltalk)中称为方法(method)，对象中的成员变量或数据项在 Smalltalk 中称为实例变量(instance variable)，调用对象的成员函数称为发送消息给对象(sending message)。

用面向对象的方法来解决问题，不再将问题划分为解决它的函数，而是将问题细分为对象。在真实世界中，这些对象可能是用户定义的数值数据类型(如复数)、数据库(如人事文件)、程序结构(如二元树)以及其他如计算机游戏中的角色等。

第 2 章　C++中的基本 I / O 语句

2.1　基本程序结构

在讨论 C++程序结构之前，先看看以下程序，它说明了简单、具体的 C++程序结构。

2.1.1　字符串的打印

cout 打印字符串的方式和 printf( )函数类似，只需将要打印的字符串放在双引号(″ ″)内就可以了。

实例: ch2__1.cpp

```
// ====== Program Description ======
//   program name: ch2__1.cpp
//   Using cout for printing
// ================================
#include <iostream.h>
void main( )
{
    cout << "Introduction to OOP language.";
    cout << "Introduction to OOP language.";
}
```

输出:

```
Introduction to OOP language. Introduction to OOP language.
```

实例: ch2__2.cpp

```
// ====== Program Description ======
//   program name: ch2__2.cpp
//   Using cout for printing
// ================================
#include <iostream.h>
void main( )
{
    cout << "Introduction to OOP language.\n";
    cout << "Introduction to OOP language.\n";
}
```

输出:

```
Introduction to OOP language.
Introduction to OOP language.
```

其实，同一个 cout 操作符也可用于输出多个字符串，其规则如下:

```
cout << 字符串 1 << 字符串 2 << … << 字符串 n;
```

或分成多行输出:

```
cout  <<  字符串1
```

```
        << 字符串2
           :
        << 字符串n;
```

实例：ch2__3.cpp

下面是用一个 cout 输出多个字符串的应用程序：

```
// ====== Program Description ======
//   program name: ch2__3.cpp
//   Print strings using cout.
// ================================
#include <iostream.h>
void main( )
{
    cout << "Introduction to OOP language.\n"
         << "Introduction to OOP language.\n";
    cout << "The" << " JX " << "Press";
}
```

输出：

```
Introduction to OOP language.
Introduction to OOP language.
The JX Press
```

2.1.2 整数的输出

cout 在用来输出整数时，其使用格式如下：

```
cout << 整数变量 1 << 整数变量 2 << … << 整数变量 n;
```

或写成：

```
cout  << 整数变量1
      << 整数变量2
         :
      << 整数变量n;
```

在输出整数数据时，cout 会预留恰当的空间供其使用。

实例：ch2__4.cpp

下面是一个用 cout 输出整数的应用程序：

```
// ====== Program Description ======
//   program name: ch2__4.cpp
//   Using cout for integer output.
// ================================
#include <iostream.h>
void main( )
{
    int i,  j;
    i = 10;
    j = 15;

    cout << i << " + " << j
         << " = "
```